Universidade Tecnológica Federal do Paraná – UTFPR
Departamento Acadêmico de Eletrônica – DAELN
Departamento Acadêmico de Informática – DAINF
Computer Engineering
Integration Workshop 3 (EEX23) – S71 – 2022/1

Technical Report
Handy Wardrobe

Alan Jun Kowa Onnoda – alanonnoda@alunos.utfpr.edu.br
Bruna Araujo Pinheiro – brunap@alunos.utfpr.edu.br]
Gabriel Requião – grequiao@alunos.utfpr.edu.br
Marina Pereira de Souza – marinasouza@alunos.utfpr.edu.br
Viviane Lima Bonfim Moroni de Souza – vivianebonfim@alunos.utfpr.edu.br

June, 2022

Abstract

The way a person dress may imply personal features like their personality, beliefs, or even social class. So when it comes to blind or visually impaired people, they may encounter some problems in dressing themselves due to their disability. It may take a lot of caution and organization or they have to depend a lot on sighted people to manage their clothes. With this in mind, this project implemented a smart wardrobe system which can automatically detect certain colors and patterns of clothing, as well as having several accessibility options such as having a complete inventory of it or bringing a requested piece of clothing in order to help and improve blind people quality of life and solve those problems.

1 Introduction

Blind and visually impaired people can face distinct difficulties in their life and one of those could be the inability to recognize colors and patterns of clothing they own. This can lead to many tough situations such as wearing an inappropriate clothing to the occasion, or even have their quality of life declining due to such issues or inability to enjoy following fashion trends.

Thus, the idea of this project is to solve this problem and improve their life by building a wardrobe that is able to recognize, store, manage and retrieve pieces of clothing at the command of the user through a mobile app that is designed to be accessible to blind people, making it so that they are less reliant on other people for such things.

However, due to the impractically of building a full-sized wardrobe and present it, the project has a smaller proof of concept wardrobe that should work when it is scaled up.

1.1 Overview

Figure 1 shows the Block Diagram of the system. As can be seen, the entire system is centralized around an embedded system controlling the several functions of the system and being connected with a mobile app which will serve as the user interface.

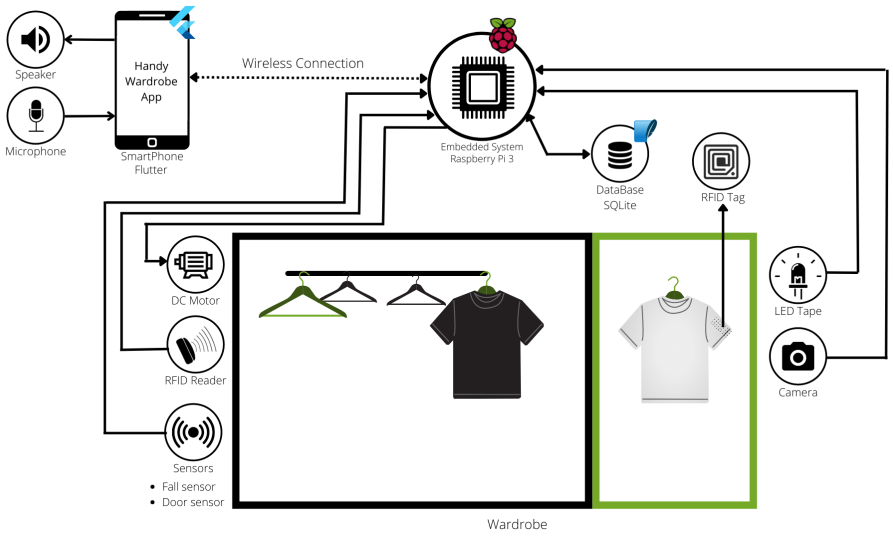

Figure 1: Block Diagram

The embedded system consist of a Raspberry Pi 3B+ connected to several components, which are as follows, a camera and LED Tapes in a closed off section of the Wardrobe so that the system can recognize colors and patterns of pieces of clothing, as well as a magnetic door sensor so that the system can know when to take the photos. A geared DC Motor and rotating mechanism are necessary so that the wardrobe can store and find the clothing as requested by the user, the rotation mechanism is a modified plastic kitchen gas cylinder holder that is directly attached to the DC Motor, with some additional modifications in order to prioritize the security of the blind person.

The system also has a RFID sensor so that it can recognize each clothing by it's unique ID, allowing it to know which clothing the user has put in the wardrobe as well as bring it to them when requested. Additionally, there's two Infrared Sensors that are responsible for the detection of fallen clothing.

The app serves as the control panel of the entire Wardrobe, sending requests to the Raspberry Pi through *REST* requests, which is running a web server with a local database that keeps tracks of the situation in the wardrobe. Since this is primarily intended for accessibility for Blind People, the app must be compatible with the operating system accessibility options for text-to-speech, as well as provide options for the user to record audio descriptions of the clothing and play it back.

To note, the wardrobe and all mechanical components are not to be built as a full-sized one but a smaller proof of concept that works on baby clothing.

2 Project Specification

The project was separated into three main components, the Wardrobe, the Embedded System and the App, with their requirements described in the sections 2.1, 2.2 and 2.3 respectively.

2.1 Wardrobe Requirements

2.1.1 Functional Requirements

W-FR01 - The Wardrobe must allow the user to control it through the App:
W-FR01.01 - The Wardrobe must allow the user to verify its inventory.
W-FR01.02 - The Wardrobe must allow the check-in of a T-shirt on the system.
W-FR01.03 - The Wardrobe must allow the check-out of a T-shirt from the system.
W-FR01.04 - The Wardrobe must register the T-shirt profile in the App.
W-FR01.05 - The Wardrobe must allow the user to select a T-shirt in the App for withdrawal.
W-FR02 - The Wardrobe must recognize T-shirts' patterns.
W-FR02.01 - The Wardrobe must recognize plain T-shirts.
W-FR02.02 - The Wardrobe must recognize striped T-shirts, with horizontal or vertical stripes.
W-FR02.03 - The Wardrobe must recognize plaid T-shirts, with horizontal and vertical stripes.
W-FR02.04 - The Wardrobe must recognize the T-shirt's colors, except the green screen color.
W-FR02.05 - The Wardrobe must identify T-shirts that dont fit in the categories above.
W-FR03 - The Wardrobe must have a T-shirt hanging section.
W-FR03.01 - The Wardrobe must have a cloth rack.
W-FR03.02 - The Wardrobes cloth rack must rotate.
W-FR03.03 - The Wardrobe must have space for ten (10) cloth hangers.
W-FR04 - The Wardrobe must have a T-shirt registration section.
W-FR04.01 - The Wardrobe must have a section where the unknown T-shirts can be registered.
W-FR05 - The Wardrobe must have a pick-up point to deliver a T-shirt to the user when requested.
W-FR05.01 - The Wardrobe must bring the T-shirt selected by the user to the pick-up point.

W-FR05.02 - The Wardrobe must always bring an empty cloth hanger to the pick-up point when it finishes an operation unless it is full.

W-FR06 - The Wardrobe must not present safety risks in its usability.

W-FR07 - The Wardrobe must handle all the T-shirts operations through the App.

W-FR08 - The Wardrobe must inform the user if a T-shirt falls off the rotating cloth rack.

2.1.2 Non-Functional Requirements

W-NFR01 - The Wardrobe must have a power supply.

W-NFR01.01 - The Wardrobes power supply must have a tension regulator.

W-NFR02 - The Wardrobe must perform the T-shirts detection with RFID technology.

W-NFR02.01 - The Wardrobe must have an RFID sensor.

W-NFR02.02 - The Wardrobe must manage T-shirts that have RFID tags on their sleeves.

W-NFR03 - The Wardrobe must be able to rotate the cloth rack.

W-NFR03.01 - The Wardrobes rotating cloth rack must be built with a bicycle chain of 140 cm.

W-NFR03.02 - The Wardrobes rotating cloth rack must be built with a bicycle chainring.

W-NFR03.03 - The Wardrobes rotating cloth rack must be built with PVC pipes.

W-NFR03.04 - The Wardrobes rotating cloth rack must be built with circles made of wire for the cloth hangers.

W-NFR03.05 - The Wardrobes rotating cloth rack must be controlled by a DC motor.

W-NFR03.06 - The Wardrobe must have two infrared sensors.

W-NFR03.07 - The Wardrobes rotating cloth rack must have a space of up to 15 cm between each cloth hanger.

W-NFR04 - The Wardrobes registration section must be appropriated for the operation.

W-NFR04.01 - The Wardrobes registration section must have its walls in green screen color.

W-NFR04.02 - The Wardrobes registration section must have LED tapes to illuminate the T-shirt.

W-NFR04.03 - The Wardrobes registration section must have a camera to take a picture of the T-shirt.

W-NFR04.04 - The Wardrobes registration section must have a green screen-colored pin for the cloth hanger.

W-NFR04.05 - The Wardrobes registration section must have a green screen-colored cloth hanger.

W-NFR05 - The Wardrobe must be constructed with Medium Density Fiberboard (MDF).

W-NFR05.01 - The Wardrobes width must be a maximum of 115 cm.

W-NFR05.02 - The Wardrobes height must be a maximum of 65 cm.

W-NFR05.03 - The Wardrobes depth must be at a maximum of 60 cm.

W-NFR05.04 - The Wardrobes registration section door must have a magnetic door sensor.

W-NFR06- The Wardrobe must isolate dangerous components from the user.

W-NFR06.01 - The Wardrobe must not present sharp edges.

W-NFR06.02 - The Wardrobes DC motor must be protected.

W-NFR06.03 - The Wardrobes power supply must be protected.

W-NFR07 - The Wardrobe must handle T-shirts that weigh 80 grams at maximum.

W-NFR09 - The Wardrobe must have a Raspberry Pi to control its Embedded system.

W-NFR10 - The Wardrobe must only have green screen-colored cloth hangers.

2.2 Embedded System Requirements

2.2.1 Functional Requirements

E-FR01 - The Embedded System must communicate with the App

E-FR02 - The Embedded System must execute a T-shirt registration process.

E-FR02.01 - The Embedded System must capture the T-shirts image.

E-FR02.02 - The Embedded System must process the T-shirts image.

E-FR02.03 - The Embedded System must collect the information about the T-shirt.

E-FR02.04 - The Embedded System must store the information about the T-shirt.

E-FR02.05 - The Embedded System must inform the App when the T-shirts registration is finished.

E-FR03 - The Embedded System must control the Wardrobes elements.

E-FR03.01 - The Embedded System must be able to control the rotating cloth rack.

E-FR03.02 - The Embedded System must be able to use the RFID sensor and the rotating cloth rack to identify a T-shirt for withdrawal.

E-FR03.03 - The Embedded System must control the rotating cloth rack to move the T-shirt requested for withdrawal to the pick-up position.

E-FR03.04 - The Embedded System must be able to detect if the registration section door is closed.

2.2.2 Non-Functional Requirements

E-NFR01 - The Embedded System must have a Raspberry Pi.

E-NFR02 - The Embedded System must be responsible for the image processing.

E-NFR02.01 - The Embedded System must use OpenCV's Histogram of Oriented Gradients (*HOG*) function in order to identify the T-shirt's patterns.

E-NFR03 - The Embedded System must use a local web connection to communicate with the App.

E-NFR03.01 - The Embedded System must send messages to the App using SSE (Server Sent Events).

E-NFR03.02 - The Embedded System must receive messages from the App using REST API.

E-NFR03.03 - The Embedded System must use Wi-Fi and access point.

E-NFR04 - The Embedded System must have firmware software to control the Wardrobes hardware.

E-NFR04.01 - The Embedded System must control the 5MP Raspberry Pi camera placed inside the registration section to photograph the T-shirt that is being registered.

E-NFR04.02 - The Embedded System must drive the DC motor to control the rotating cloth rack.

E-NFR04.03 - The Embedded System must interface with the magnetic door sensor to check if the Wardrobes registration section door is closed.

E-NFR04.04 - The Embedded System must interface with the infrared sensors to detect if a T-shirt falls.

E-NFR04.05 - The Embedded Systems firmware must be developed in Python.

E-NFR05 - The Embedded System must have a server hosted in the Raspberry Pi.

E-NFR05.01 - The Embedded System server must be developed in Flask Python.

E-NFR05.02 - The Embedded System server must use an SQLite database.

2.3 App Requirements

2.3.1 Functional Requirements

A-FR01 - The App must start the Wardrobes functions.

A-FR01.01 - The App must allow the user to register a new T-shirt placed in the registration section in the Wardrobe.

A-FR01.02 - The App must allow the user to check-out a specific T-shirt.

A-FR01.03 - The App must allow the user to check the T-shirts inventory.

A-FR01.04 - The App must allow the user to request a T-shirt for withdrawal by color or pattern.

A-FR01.05 - The App must request that the user chooses one option if there are two or more T-shirts are matching the request made for withdrawal.

A-FR02 - The App must communicate with the Wardrobe.

A-FR03 - The App must inform the user of important actions that should be taken using the smartphones speakers.

A-FR03.01 - The App must announce that the T-shirt must be placed in the Wardrobes registration section.

A-FR03.02 - The App must announce when the T-shirts registration has ended.

A-FR03.03 - The App must announce that the user may remove the T-shirt from the Wardrobes registration section.

A-FR03.04 - The App must announce that the user may place the recently registered T-shirt in the Wardrobe.

A-FR03.05 - The App must announce if a T-shirt has been put incorrectly in the wardrobe and request correction.

A-FR03.06 - The App must announce if a T-shirt fell in the Wardrobe or if it is missing.

A-FR03.07 - The App must announce that the Wardrobe is operating the rotating cloth rack and that the user should keep a safe distance from it.

A-FR04 - The App must allow the user to record audio files to complement a T-shirts description.

A-FR05 - The App must relay user commands to the Wardrobe Embedded system.

A-NFR06.01 - The App must send a message to the Wardrobes Embedded system server to start the registration process of a T-shirt.

A-NFR06.02 - The App must receive a message from the Wardrobes Embedded system server to end the registration process of a T-shirt.

A-NFR06.03 - The App must request information from the Wardrobes Embedded system server.

2.3.2 Non-Functional Requirements

A-NFR01 - The App must be developed using Flutter.

A-NFR01.01 - The App must be developed using the language Dart.

A-NFR01.02 - The Apps Flutter code must follow good programming practices to make it accessible.

A-NFR02 - The App must communicate with the Wardrobes Embedded system.

A-NFR02.01 - The App must send messages to the Wardrobes Embedded system server using the REST API.

A-NFR02.02 - The App must receive messages from the Wardrobes Embedded system server by SSE (Server Sent Events).

A-NFR03 - The App must use the smartphone speakers to make announcements.

A-NFR04 - The App must be compatible with the smartphones accessibility tools.

A-NFR05 - The App must be in Brazilian Portuguese.

3 Development

This section describe the process of the development of the project, explaining technical details as well as difficulties. Further diagrams can be seen in the project's blog [1].

3.1 Mechanical Structure

Development started with the design and creation of the mechanical diagrams using the Autodesk AutoCAD [2], which is an industry standard tool for mechanical design, the resulting diagram can be seen in Figure 2. The Wardrobe was built using MDF(*Medium Density Fiberboard*).

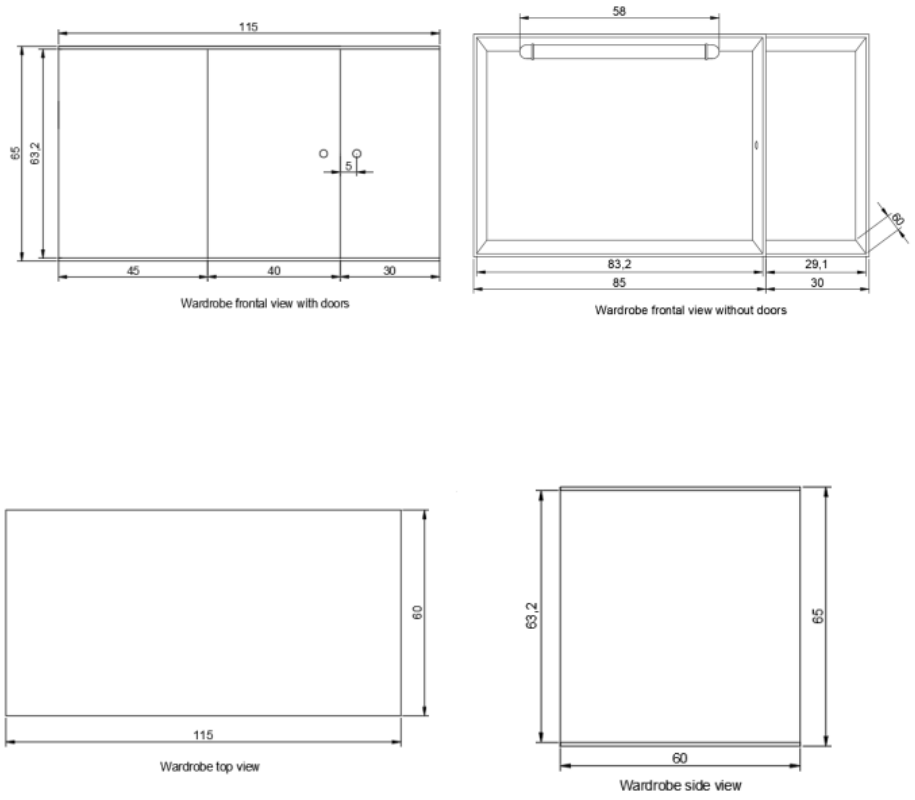

Figure 2: Mechanical Schematics

The initial plan for the rotating rack mechanism can be seen at Figure 3, which consisted of four PVC pipes, with parts of them being cut off so that the

clothing hanger can be put and a bicycle chain with 3 gears(including the motor one) driving it.

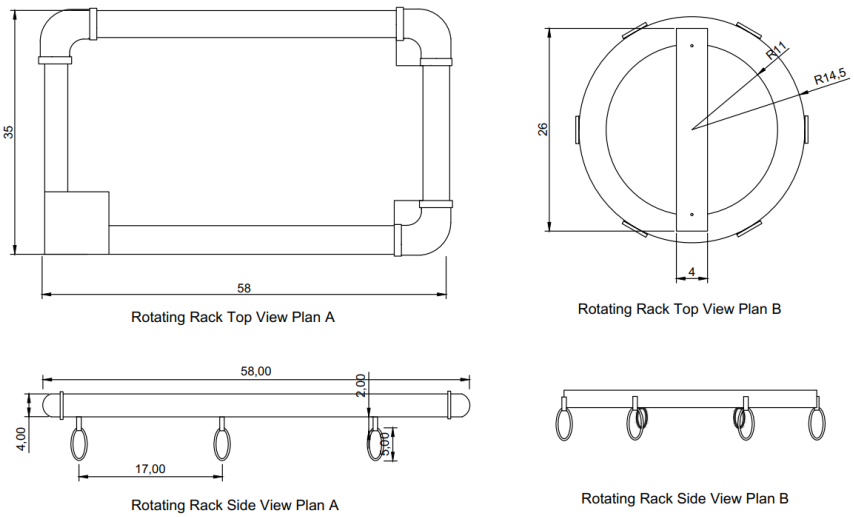

Figure 3: Rotating Rack Mechanism Plan A and B

This plan however did not work and so the decision was made, as per the risk analysis plan dictated, to switch to the plan B as shown in its built form in the Figure 4, which consists of a plastic kitchen gas cylinder holder that is modified with places to put the clothing hanger as well as a piece of MDF where the DC motor connects to it.

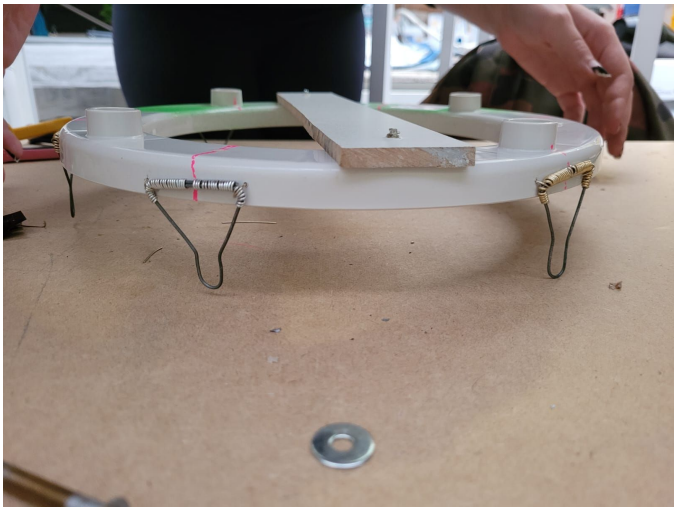

Figure 4: Rotating Rack mechanism built

3.2 Hardware Project

The start of the Hardware Project was the drawing of the electrical schematics in the the tool Eagle [3], this diagram can be seen in Figure 5.

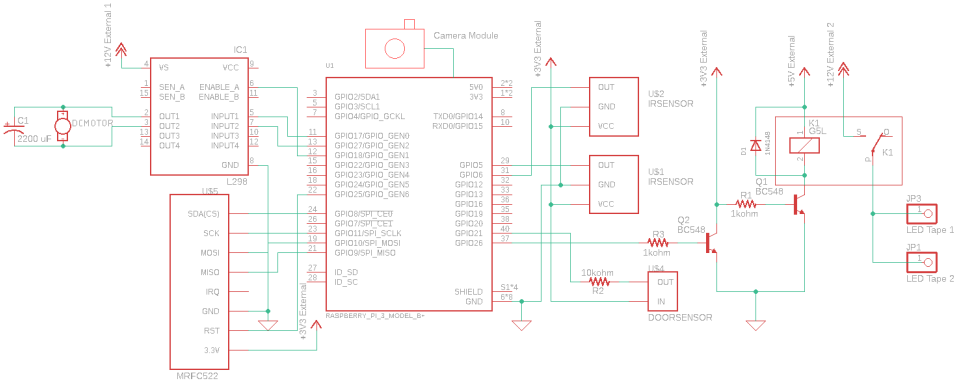

Figure 5: Hardware Schematics

From the diagram, we can see that the Raspberry Pi 3 Model B+ is the heart of the design and connects all components together. For the control of the DC Motor, it was chosen a L298 H-Bridge Module, specially due to availability and ease of connectivity, noting that the ENABLE_A pin was connected to the GPIO18 of the Raspberry Pi for the usage of one of the two Hardware RPM channels [4].

For the RFID Reader, after extensive research, it was decided to go with the MFRC522 module [5], due to price and it covering the uses cases needed. It was researched beforehand that if needed, one of the registers could be changed to increase the antenna attenuation and thus giving it some boost in range, but from the testing performed, the default configuration was already enough.

The Infrared Sensors are two simple modules with a LM339 comparator and an adjustable potentiometer. The magnetic door sensor is a simple magnetic switch, noting that we use one of the internal pull-ups of the Raspberry Pi so that it doesn't float when it's not connected.

For the activation of the LED tapes, it has been used a Relay Module with one included transistor BC548. The Raspberry Pi has trouble with that transistor alone and so it was included an additional external BC548 to boost the current and isolate the Raspberry Pi of any possible issue. The Relay module was chosen over a transistor for switching purposes so that the amount of LED Tape used could be easily changed without having to worry about how much current it is drawing.

All of the components were tested individually before integration into the final hardware project.

3.3 Computer Vision

In order to fulfill the requirements of the project, it was necessary to develop an algorithm that can detect certain colors and patterns in the image taken of the pieces of clothing. The algorithm was developed in Python using the OpenCV library [6], which is widely used for Computer Vision purposes.

In the start of the algorithm, the taken picture is cropped and resized. For the color detection, the image is transformed into it's HSV(Hue-Saturation-Value)[7] equivalent and then masks for the range of each color are applied, and the features of each of the resulting masks are counted. If there are enough features, then it is decided that such color is present in the clothing. Note that only a selected set of colors can be detected and not any arbitrary one.

For the pattern detection, the *History of Oriented Gradients*(HOG)[8, 9] was used to make the detection of the presence or lack of vertical and horizontal stripes, with the shirt being plain if none are detected or plaid if both are detected.

The Figure 6 shows some examples of the pictures taken on the registration area and the return of the algorithm for each one of them.

Figure 6: Image Processing Results

3.4 Firmware & Backend

Both the Firmware and Backend are written in Python 3.7, with the Firmware - responsible for control of all hardware components - being run in a separate thread. Communication is done through a a Python Message Queue[10]. The Firmware uses a hardware PWM library[11], the standard GPIO library of the Raspberry Pi[12] as well as a Python driver for the MFRC522 RFID Reader [5].

The Backend uses the framework Flask [13] to create the web application with the *REST* endpoints for the communication with the mobile app. The requests are processed, doing queries and changing the database if necessary, and then forwarded to the Firmware so that the wardrobe can take the necessary actions.

When the Firmware detects that one of the piece of clothing has fallen off(by the usage of the Infrared Sensors), it sends a message through one of the message queues to the Back End which then sends it to the mobile app through the usage of *Sever-Sent Events*(SSE). The library used for this purposed is called Flask-SSE [14], and requires the usage of the open-source in-memory database *Redis* [15].

The webserver used along with the Flask application is *Gunicorn WSGI HTTP Server*[16] using Gevent as the worker type, which are asynchronous workers based on gevents and cooperative multi-threading [17], using Python co-routines instead of spawning new threads for requests, allowing the Server to process a lot more requests concurrently.

The database chosen for the project is a SQLite[18], due to itss simplicity, ease of use and the very light requirements that the system has for a database solution, which is a very widely used solution for when the application requires light usage of a database. It is used alongside SQLAlchemy [18], a *Object Relational Mapper* library for usage with databases, providing the data mapper patterns where classes and objects are mapped to the database, permitting ease of development, speed, as well as added security(due to SQL injection attacks not being possible).

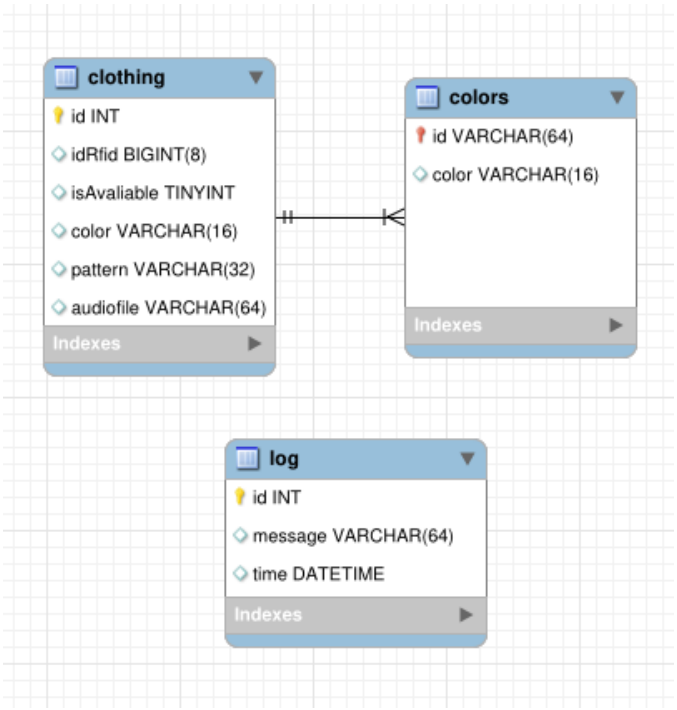

Figure 7: Entity-Relationship Diagram

The database can be seen in the Figure 7 which shows the Entity-Relationship Diagram for it. It's a simple solution that fits the needs of the project, with clothing table having a one-to-many relationship with colors table as well as a simple log class for development purposes.

3.5 Mobile App

The App was developed using Flutter framework [19] with the Dart [20] programming language. Several of the App screens can be seen in the Figure **??**. The App connects with the backend and thus the firmware by REST requests. One thing to note is that the app was designed from the grounds up to be compatible with the accessibility tools that permits a blind person to use a smarphone app.

The App permits the user to see the inventory, add and remove clothing as well as request them from the wardrobe and see the full inventory with the possibility of filtering it by colors and patterns. It also has the capability to record an audio description when adding a new clothing and play it back to the user.

As well as warning the user in case some clothing has fallen off the wardrobe.

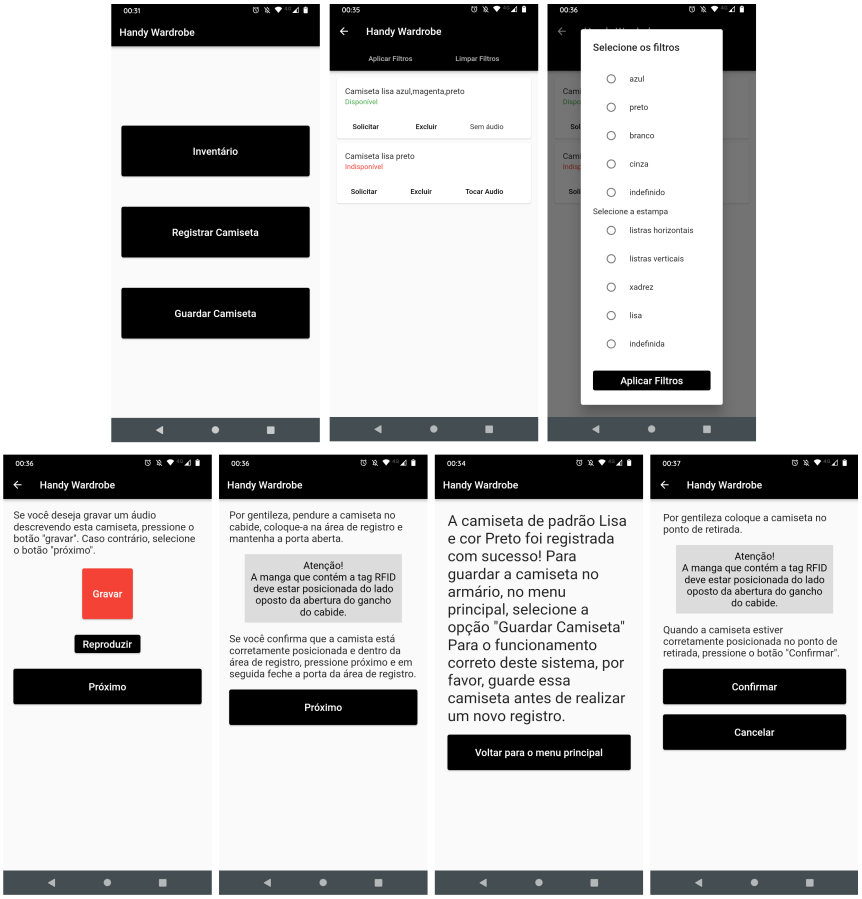

Figure 8: App Screens

In order to warn the user, the app listens to a *SSE* channel and when it receives a message, it display a warning page to the user.

4 Result

The team was able to develop a proof of concept wardrobe fulfilling all the requirements planned from the start with satisfying results as an accessibility tool aiming to improve the quality of life for blind people.

In the end, the proof of concept is able to automatically register clothing, detecting it's color and patterns, storing it and taking it out on command, all with a simple app that works well with the text to speech tools provided by the Operating System.

4.1 Budget

For the project, it was projected to have a budget about R$200 to R$250 per member of the team, therefore a total budget of R$1000 to R$1200. Table 4.1 shows all components brought and their price, the total cost was R$ 1258 which is slightly higher than the required amount.

Component	**Quantity**	**Price per unit (R$)**
Raspberry Pi Model B+	1	450
Geared DC Motor 12V	1	93
H Bridge L298N	1	25
LED Tape	1	12
RFID Tags	10	3.9
MFRC522 RFID Sensor	1	29
Power Supply	1	50
IR Sensor	2	7
Mechanical Components	1	500
Extra Expenses	1	58
Total		**1258**

Table 1: Budget

4.2 Schedule

The development time for the project was 7 weeks. Table 4.2 shows an overview of the entire schedule, noting that Estimated Hours include the 30% extra hours. The full sheet can be viewed at [21].

Deliverable	**Estimated Hours**	**Hours Worked**
Blog and Planning	132	118
Mechanical Project	147	111
Hardware Project	72	46
Software Project	77	71
Integration	110	186
Final Tests and Documentation	26	30
Total	**564**	**562**

Table 2: Project Schedule

5 Conclusion

The team faced many challenges during this project, specially due to mechanical issues, as team members had little to no knowledge in mechanics, even having it happen one of the risks that was analyzed in the Project Charter happen and the team losing a lot of time trying to fix, fail and changing it to the second plan.

Despite the many challenges, the team was able to overcome all of them while growing and learning from all the mistakes made at all steps of the development. And at the end, the project was a success with all the major components working reasonably well and within expectations.

It's important to note that the planning step was essential to this project, especially the risk analysis plan, as without it and the backup plans for it, the project might not have been concluded with success.

References

[1] Team Narnia. Handy wardrobe blog. https://lowly-cover-d2d.notion.site/The-Handy-Wardrobe-7c37a347de4848258b3834120f796e80.

[2] Autodesk. Autocad overview. https://www.autodesk.com/products/autocad/overview.

[3] Autodesk. Eagle overview. https://www.autodesk.com/products/eagle/overview.

[4] Broadcom. Bcm2835 peripherals.

[5] pimylifeup. Mfrc522 driver. https://github.com/pimylifeup/MFRC522-python.

[6] OpenCV Team. Opencv. https://opencv.org/.

[7] sentdex. Hog (histogram of oriented gradients): An overview. https://cvexplained.wordpress.com/2020/04/28/color-detection-hsv/.

[8] Satya Mallick. Histogram of oriented gradients explained using opencv. https://learnopencv.com/histogram-of-oriented-gradients/.

[9] Mrinal Tyagi. Hog (histogram of oriented gradients): An overview. https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f/.

[10] Python. A synchronized queue class. https://docs.python.org/3/library/queue.html.

[11] Pioreactor. rpi hardware pwm. https://github.com/Pioreactor/rpi_hardware_pwm.

[12] croston. raspberry-gpio-python. https://pypi.org/project/RPi.GPIO/.

[13] Flask Team. Welcome to flask. https://flask.palletsprojects.com/en/2.1.x/.

[14] David Baumgold. Flask sse. https://flask-sse.readthedocs.io/en/latest/quickstart.html.

[15] Redis Ltd. Redis. https://redis.io/.

[16] Gunicorn Team. Gunicorn. https://gunicorn.org/.

[17] Gunicorn Team. Gunicorn server model. https://docs.gunicorn.org/en/latest/design.html.

[18] SQLAlchemy. Sqlalchemy. https://www.sqlalchemy.org/.

[19] Flutter. Flutter documentation. https://docs.flutter.dev/.

[20] Dart. Dart overview. https://dart.dev/overview/.

[21] Team Narnia. Handy wardrobe schedule. https://docs.google.com/spreadsheets/d/1D0q1-g6olnMaT4FFDSA4lekVcGmIv9JdLSsdwiloOuM/.